Diakoptics, Domain Decomposition and Parallel Computing

C.-H. LAI

School of Mathematics, Statistics and Scientific Computing, University of Greenwich, Wellington Street, Woolwich, London SE18 6PF, UK

This paper gives a brief history of diakoptics and provides an insight into its connections with domain decomposition methods. The aim is to discuss some of the common grounds of the two methods and to relate these common grounds to parallel computing. Load balancing in an MIMD environment and implementation issues in an SIMD environment are discussed.

Received September 8, 1994, revised December 8, 1994

1. HISTORICAL DIAKOPTICS

In 1963, Kron published a book entitled '*Diakoptics: The Piecewise Solution of Large-Scale Systems*', which was a collection of technical papers concerning the concepts and applications of diakoptics and was written by himself and published in the Electrical Journal between 1957 and 1959. The terminology *diakoptics* was originated from the Greek words *kopto* which means to tear and *dia* which means systems. Hence diakoptics can be interpreted as system-tearing (7), and it is sometimes referred to as the method of tearing. The idea is to solve a class of system problems, typically of electrical circuits, that has a very large number of variables which is then torn into a number of subdivisions, each subdivision can be analysed and solved separately as if the other subdivisions were non-existent. Then the partial solutions are interconnected, step by step, until the solution of the entire system is accomplished.

The original purpose of the method of tearing was not to save computation, but to facilitate the solution of complex system problems that cannot be solved in one piece. The idea was used in the calculation of large and complicated electrical circuits in early 20th Century when no digital computers of significant size were available to handle a given problem in one piece. At that time, electrical engineers analysed and calculated a large number of extensive a.c. network problems. These results were considered as the building blocks for a larger and more complex network. Thus an electrical engineer was able to paste these building blocks to a desirable network as required. Each of these building blocks involved typically the inversion of a 2×2 or 3×3 matrix. However, the idea that Kron had in his mind, as discussed in the preface of the book *Diakoptics*, was not confined to electrical circuit calculations, but to search for a 'Mathematics of large-scale engineering systems'. Some work was done by Kron along this idea in as early as 1940s including, numerical solutions of differential equations (5), compressible and incompressible fluid flow fields (4), and elasticity (3). He tried to replace some continuous engineering problems by means of an equivalent electrical circuit which contains finite number of unknowns. It is clear that Kron's aim was, in modern terminology, to replace a continuous system by a discretized system, using a unified approach based on electrical networks. The idea proposed by Kron did not get across to other scientific communities, in particular the numerical analysts at that time. As it is well known, the only discretization method that was used to replace continuous problems adapted by most numerical analysts and other fluid dynamists until late 1940s was the finite difference method (2, 6). Nevertheless, Kron made a significant impact to the electrical engineering community and his idea formed one of the major techniques in solving large electrical network problems even in the early days of computers.

The concept of tearing used in the early days of computer has certainly influenced some of the contemporary computational scientists, in particular people involved in parallel computing, in the design of parallel algorithms for partial differential equations. It is believed that Kron was a pioneer in the application of, the so called, domain decomposition methods to large scale engineering problems. Other important contributions including, early tensor analysis and applications of matrices in network analysis, and development of graph theory for applications in the method of tearing. Section 2 presents the contemporary interpretation and understanding of the tearing concept. The concept is then presented in a capacitance matrix form and a direct method of solution of the resulting discrete system is discussed. An alternative method of solutions is to apply a pre-conditioned conjugate gradient type of methods which results to the iterative substructuring technique. Section 3 presents the idea of an iterative method based on the iterative coupling technique proposed by Schwarz. A number of variants of the iterative coupling technique is also included. The connection of the iterative coupling method and the concept of diakoptics is discussed. Parallel computational properties of the

above methods, load balancing and implementation issuses are considered in due course.

2. DIAKOPTICS IN THE PARALLEL COMPUTING ERA

It is rather clear from the historical background that the method of tearing is particularly suitable for present days parallel computer applications. The amount of memory in computers of early days was limited, and hence cannot be used to handle large number of discrete unknowns. The situation is similar to the present days parallel computers that every processing element has a finite amount of local memory. It is also similar to the present days distributed computing systems where each of the loosely coupled computers has a limited amount of local memory. Although some MIMD nodes, such as transputers, can offer up to 4MBytes of local memory (22), it is the ever increasing demand in computational power and memory requirement for large scale numerical simulations such as solidification processes, oceanography, and unsteady aerodynamics using computational fluid dynamics techniques (19, 29) that diakoptics is brought back as a parallel algorithm. It was used in a sequential fashion during 1940s, while it is used in a concurrent fashion at present since each of the subdivisions described above can be computed concurrently. For simplicity, if the number of subdivisions is the same as the number of processing elements in a coarse-grained parallel environment, then each subdivision will follow exactly the same algorithm as if it was solved on a sequential computer. Investigation on transputers can be seen in Bowden's work (20). However, the finite number of subdivisions can also be run on a SIMD massively parallel processing environment. In this case, one has to solve each of the subdivisions on the parallel environment and a SIMD parallel algorithm is required to solve this subdivision. Work in this aspect can be found in Wait (15). The interconnection method, or in modern terms numerical coupling of subproblems, was not mentioned in Diakoptics. It is indeed the major issue that has been left open, and until recently numerical mathematicians are still developing accelerated coupling techniques. One reason for such research is because direct methods are computationally expensive for large problem size, and hence recent effort has been concentrated on the development of iterative techniques. In fact, the concept of diakoptics as presented in (7) involves only the tearing and a coupling technique with a direct method of solutions to the coupling technique.

A philosophical introduction to Kron's work and its parallel implication was given by Bowden in a one day symposium entitled 'Alternative Models of Computation and New Routes to Parallelism', organized by the British Computer Society Parallel Processing Specialist Group and the Cybernetic Machine Specialist Group was held at the University of Greenwich on 12th March, 1994 (27).

2.1. Capacitance matrix representation of tearing

Now instead of working with an electrical network, a model second order, positive definite, self-adjoint elliptic Dirichlet problem on a bounded domain R^2 with piecewise smooth boundary is considered. A second order accurate finite volume method is applied to discretise the continuous problem which leads to the linear system

$$Au = f$$

where A is a symmetric and positive definite $n \times n$ matrix, $u = (u_1 u_2 \cdots u_n)^T$ are the discretised unknowns, and $f = (f_1 f_2 \cdots f_n)^T$. Suppose an artificial interface is introduced inside the domain, and that the unknowns are partitioned into two parts, u_γ and u_d, where u_γ denotes the nodal function values defined on the nodal points along the interface and u_d denotes the nodal function values defined on all other nodal points. Then the above system can be rearranged as,

$$\begin{pmatrix} A & G \\ G^T & B \end{pmatrix} \begin{pmatrix} u_d \\ u_\gamma \end{pmatrix} = \begin{pmatrix} f_d \\ f_\gamma \end{pmatrix}$$

where $u = (u_d u_\gamma)^T, f = (f_d f_\gamma)^T$. Consequently, by means of a Gaussian elimination, one obtains the capacitance system consists of the unknown u_γ,

$$Cu_\gamma = g$$

where $C = B - G^T A^{-1} G$ and $g = f_\gamma - G^T A^{-1} f_d$. The terminology capacitance was introduced by Hockney (9) in the context of imbedding an irregular region in a regular one for electrostatic problems. Such imbedding is also known as fictitious domain methods (11). The capacitance matrix method was also investigated by Buzbee *et al.* (10) and Dryja (12) in the context of dividing an irregular domain into a number of non-overlapped but regular subdomains. The matrix C, known as the Schur complement of B, is also symmetric and positive definite. In the context of diakoptics, the field problem is discretized by means of an electrical network analog of which the discretized system is the same as that given above. If the network analog is partitioned into two portions, then a rearranged system as the one given above is obtained.

In order to solve the capacitance matrix equation, one can either construct the matrix C and the vector g or use a preconditioned conjugate gradient method. The construction of matrix C is known as Diakoptics and is expensive on a sequential machine, but was considered by Wait on a fine-grained SIMD environment (15) and by Bowden on a coarse-grained message-passing environment (20) and more recently by Merugu and Fusco (26) for some electromagnetic field problems. For simplicity, it is assumed that the interface separates the domain into two nonoverlapped subregions. Therefore A is a 2×2 block diagonal matrix denoted as

$$A = \begin{pmatrix} A_{11} & 0 \\ 0 & A_{22} \end{pmatrix}$$

and

$$f_d = \begin{pmatrix} f_{d1} \\ f_{d2} \end{pmatrix}$$

The most computationally intensive part of solving the capacitance matrix system is the construction of $A^{-1}f_d$ which involves the solution of the two subproblems $A_{11}u_{d1} = f_{d1}$ and $A_{22}u_{d2} = f_{d2}$, where $u_d = (u_{d1}u_{d2})^T$.

There are two different parallel implementations for the solution of the two subproblems. First, assuming a fine-grained SIMD environment, a data parallel technique for the solutions of subproblems is required, and these subproblems are solved sequentially. Each subproblem should be easily mapped onto the fine-grained environment. Second, solving the two subproblems concurrently in a 2-processing element message-passing environment, and a serial subproblem solver is required. Each subproblem should have similar work load so that the algorithm suits a coarse-grained environment. Similarly the construction of $A^{-1}G$, where G contains as many columns as the number of unknowns along the interface, requires the same attention on work load. As the discretised problem size increases, the amount of work to obtain A_{11}^{-1} and A_{22}^{-1} increases. Therefore such work eventually outweighed the overheads involved in communication of a message-passing environment, and hence it becomes a more efficient algorithm. However, as the number of subdomains increases with the same problem size, G then contains more columns which means that $A^{-1}G$ involves more and more subproblem solves. Therefore it is worth to store a Cholesky factorization of A so that $A^{-1}G$ can be performed more effectively. Also, the calculation of A^{-1} involves A_{11}^{-1}, A_{22}^{-1}, A_{33}^{-1}, ..., etc. which can be performed concurrently in a message-passing environment. Since the problem size is fixed, increasing the number of subdomains amounts to decrease the matrix size of A_{11}^{-1}, A_{22}^{-1}, ..., etc. The inverses are computationally inexpensive but at the cost of increasing overheads due to communications. Such computation becomes expensive for nonlinear problems where the matrix A is different for every Newton's iteration. The other method is to use a preconditioned conjugate gradient solver for the system $Cu_\gamma = g$ as discussed in next section.

2.2. Iterative substructuring methods

The first paper which described a network analogy of elastic structures can be found in Kron's work (3). Applications of tearing in the context of elastic structures were used in a similar way as that in an electrical network, and the idea is equivalent to the substructuring idea first mentioned by Przemieniecki in (8). Classical work includes Dryja (12), Golub and Mayers (13), recent work by Bjorstad and Widlund (16), all involved in applying a preconditioned conjugate gradient algorithm to the capacitance matrix system. Similar research are considered more recently by many researchers in domain decomposition, linear algebra, and parallel computing research communities. Work to solve the original system by means of an iterative method was given by Pasciak, and Schatz (17). Instead of using the elimination process as discussed above, preconditioned conjugate gradient methods were used to solve the original system

$$\begin{pmatrix} A & G \\ G^T & B \end{pmatrix} \begin{pmatrix} u_d \\ u_\gamma \end{pmatrix} = \begin{pmatrix} f_d \\ f_\gamma \end{pmatrix}$$

which is then written as

$$\begin{pmatrix} A_{11} & 0 & G_{13} \\ 0 & A_{22} & G_{23} \\ G_{13}^T & G_{23}^T & B \end{pmatrix} \begin{pmatrix} u_{d1} \\ u_{d2} \\ u_\gamma \end{pmatrix} = \begin{pmatrix} f_{d1} \\ f_{d2} \\ f_\gamma \end{pmatrix}$$

for two nonoverlapped subdomains. The equivalence of using preconditioned conjugate gradient methods for the original system and for the capacitance system is due to Eisenstat and was discussed by Keyes and Gropp (18). Suppose a preconditioned conjugate gradient method is applied to $Cu_\gamma = g$ with initial iterate $u_\gamma^{(0)}$ and preconditioner M, it is equivalent to another preconditoned conjugate gradient method applied to the original system $Au = f$ with the initial iterate

$$u^{(0)} = \begin{pmatrix} A^{-1}[f_d - Gu_\gamma^{(0)}] \\ u_\gamma^{(0)} \end{pmatrix}$$

and the preconditioner

$$M_I = \begin{pmatrix} A & G \\ G^T & M + G^T A^{-1} G \end{pmatrix}$$

Since the method described here is equivalent to that described in section 2.1, the multi-subdomain implementation exhibit similar properties as that discussed above.

3. THE SCHWARZ ALTERNATING METHOD

So far, the discussion has been restricted to discretized problems. There has been a lot of work involved in the subdivision of a physical problem into a number of subproblems. For example, viscous–inviscid coupling, composite material, fictitious domain methods, two-phase flow problems etc. In these examples, the common practice is to subdivide the physical domain into a number of subdomains according to the physical properties of the governing equations or the material properties of the given domain. The decomposition is said to be done at the level of the continuous problem instead of the discretized problem. It is believed that the first attempt to provide an algorithm for continuous problems posed in an irregular domain was due to Schwarz (1), and the method is formally known as the Schwarz alternating method when Picard applied it to some nonlinear elliptic boundary value problems (24). Schwarz considered an elliptic boundary value problem posed on an irregular domain, which was then divided into two regular subdomains where solutions for each of these subdomains are available. The underlying algorithmic idea is an iterative coupling technique for the

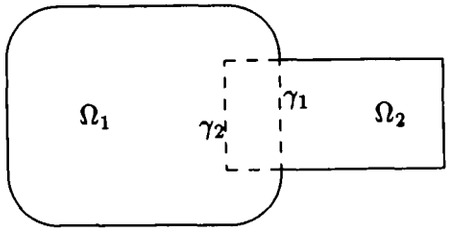

FIGURE 1. A flask-shaped region similar to that used by Schwarz

coupling of two subdomains, each of regular shape, obtained by partitioning the original domain which is of irregular shape. The following algorithm describes the Schwarz alternating method for the differential problem $Lu = f$ defined in Ω which is partitioned into two overlapped subdomains Ω_1 and Ω_2 (see Figure 1) with $f|_{\Omega_i} = f_i$, $i = 1, 2$, and the problem is prescribed with Dirichlet boundary conditions g_1 and g_2 along $\partial\Omega_1 \bigcap \partial\Omega$ and $\partial\Omega_2 \bigcap \partial\Omega$ respectively.

The Schwarz alternating method:
begin {
$n := 0$; $u_2^{(0)}|_{\gamma_1} :=$ initial approximation;
repeat {
$n := n + 1$;
$u_1^{(n)} :=$ { solve $Lu_1^{(n)} = f_1$ in Ω_1
subject to
$u_1^{(n)} = g_1$ on $\partial\Omega_1 \bigcap \partial\Omega$
$u_1^{(n)}|_{\gamma_1} = u_2^{(n-1)}|_{\gamma_1}$ };
$u_2^{(n)} :=$ { solve $Lu_2^{(n)} = f_2$ in Ω_2
subject to
$u_2^{(n)} = g_2$ on $\partial\Omega_2 \bigcap \partial\Omega$
$u_2^{(n)}|_{\gamma_2} = u_1^{(n)}|_{\gamma_2}$ };
} until converge; } end.

Here the superscript (n) denotes the number of Schwarz iterations. It can be seen that the algorithm is essentially a block iterative method where each block is a subdomain. In particular the algorithm is essentially a subdomain Gauss–Seidel iterative method. Therefore it is intrinsically sequential without suitable modification. However, one can apply the algorithm on a massively parallel computer such as the DAP or the Connection Machine without making any modification, provided that a data parallel *solve* is available. It can be seen that a data parallel direct solver or a data parallel iterative solver is required. Intensive research in this topic has been conducted by various people in late 70s early 80s which lead to various parallel algorithmic development for the numerical solutions of partial differential equations. The subdomain itervative methods were developments related to SIMD parallel algorithms during the 80s (14, 21). Although many of these parallel methods were developed independently, it is now becoming clear that the Schwarz alternating method provides a unified algorithmic framework.

Before some variants of the algorithm are presented, the discretized representation of the algorithm is described. Using the notation introduced in section 2.2, the Schwarz alternating algorithm can be written as the matrix representation,

$$\begin{pmatrix} A_{11} & G_{13} \\ G_{13}^T & B \end{pmatrix} \begin{pmatrix} u_{d1} \\ u_\gamma \end{pmatrix} = \begin{pmatrix} f_{d1} \\ f_\gamma - G_{23}^T u_{d2} \end{pmatrix}$$

$$\begin{pmatrix} B & G_{23}^T \\ G_{23} & A_{22} \end{pmatrix} \begin{pmatrix} u_\gamma \\ u_\gamma - G_{13}^T u_{d1} \end{pmatrix} = \begin{pmatrix} f_\gamma \\ f_{d2} \end{pmatrix}$$

A block Gauss–Seidel iterative method can be applied to the above matrix systems. Eliminating u_{d1} and u_{d2} leads to the same capacitance matrix system as that given in section 2.1. Hence the Schwarz alternating method is equivalent to the method of tearing. The difference between the Schwarz alternating method and Diakoptics is that the former is a block Gauss–Seidel iterative method and the latter is a direct method applied to the capacitance matrix system.

3.1. The subdomain iterative method

In Schwarz's original work, the maximum principle was used in the *solve* part of the algorithm. Computationally, one can first of all triangulate the region Ω and secondly replace the *solve* part by an *iterative solve*.

The subdomain iterative method:
begin {
$n := 0$; $u_2^{(0,0)}|_{\gamma_1} :=$ initial approximation;
repeat {
$n := n + 1$;
$\mu := 0$;
$u_1^{(n,\mu)} :=$ { iterative solve $Lu_1^{(n,\mu)} = f_1$ in Ω_1
subject to
$u_1^{(n,\mu)} = g_1$ on $\partial\Omega_1 \bigcap \partial\Omega$
$u_1^{(n,\mu)}|_{\gamma_1} = u_2^{(n-1,\mu)}|_{\gamma_1}$ };
$\mu := 0$;
$u_2^{(n,\mu)} :=$ { iterative solve $Lu_2^{(n,\mu)} = f_2$ in Ω_2
subject to
$u_2^{(n,\mu)} = g_2$ on $\partial\Omega_2 \bigcap \partial\Omega$
$u_2^{(n,\mu)}|_{\gamma_2} = u_1^{(n,\mu)}|_{\gamma_2}$ };
} until converge; } end.

Here the second superscript μ denotes the number of inner iterations. In particular when $\mu = 1$ and the overlapped region, i.e. $\Omega_1 \bigcap \Omega_2$, is of one cell thick, then the subdomain iterative method is equivalent to apply the same stationary iterative method to the re-arranged system in section 2.2. Therefore the algorithm can be formulated in a capacitance matrix approach with each subproblem being solved by means of a stationary iterative method (21). The case when $\mu = 1$ is particularly important during actual implementation on multiprocessors in order to keep minimal changes to the original code. Since the sequential algorithm is not changed, the load-balancing between the two subdomains is the major concern in order to reduce the overall computational work (23, 28). In general, a data parallel direct method on a SIMD environment is not as efficient as a data parallel iterative

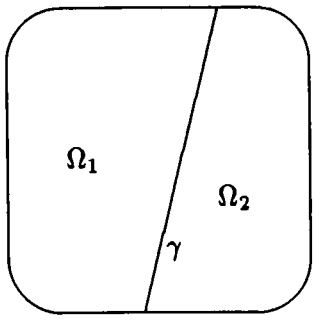

FIGURE 2. A nonoverlapped decomposition.

method (14). Therefore the subdomain iterative method was introduced as a variant of the Schwarz alternating method on a SIMD environment.

Provided that the inner iterative solve is the same as that being used in the global domain, the subdomain iterative method has the following general behaviour. First, the number of Schwarz iterations n decreases as the overlapped region increases and second, n decreases as the number of inner iterations μ increases. An optimal algorithm depends on the choice of μ and a balanced work load between subdomains.

3.2. Non-overlapped domain decomposition

It is sometimes problematic, as far as data structure is concerned, in an overlapped approach. In particular when the meshes in different subdomains do not match each other or perhaps one is finer than the other. The problem can be more easily dealt with by means of a nonoverlapped approach. Two variants of the Schwarz alternating method are presented below.

A gradient variant of the
Schwarz alternating method:
begin {
$n := 0$; $\lambda^{(0)} :=$ initial approximation;
repeat {
$u_1^{(n)} :=$ { solve $Lu_1^{(n)} = f_1$ in Ω_1
subject to
$u_1^{(n)} = g_1$ on $\partial\Omega_1 \bigcap \partial\Omega$
$\frac{\partial u_1^{(n)}}{\partial n_1} = \lambda^{(n)}$ on γ };
$u_2^{(n)} :=$ { solve $Lu_2^{(n)} = f_2$ in Ω_2
subject to
$u_2^{(n)} = g_2$ on $\partial\Omega_2 \bigcap \partial\Omega$
$\frac{\partial u_2^{(n)}}{\partial n_2} = \lambda^{(n)}$ on γ };
$\lambda^{(n+1)} := \lambda^{(n)} + \alpha(u_2^{(n)} - u_1^{(n)})|_\gamma$ };
$n := n + 1$;
} until converge; } end.

Here γ denotes the interface of the two non-overlapped subdomains as depicted in Figure 2 and n_1 and n_2 denote the outward normals along γ with respect to Ω_1 and Ω_2. The modified method can be easily implemented on a fine-grained machine provided an efficient fine-grained *solve* is available. It can also be easily implemented on a coarse-grained machine, because each subproblem is solved independently and a simultaneous update along the interface is used.

A Neumann–Dirichlet variant of the
Schwarz alternating method:
begin {
$n := 0$; $u_1^{(0)}, u_2^{(0)} :=$ initial approximation;
repeat {
$n := n + 1$;
$u_1^{(n)} :=$ { solve $Lu_1^{(n)} = f_1$ in Ω_1;
subject to
$u_1^{(n)} = g_1$ on $\partial\Omega_1 \bigcap \partial\Omega$
$u_1^{(n)} = u_1^{(n-1)} + \alpha(u_2^{(n-1)} - u_1^{(n-1)})$ on γ };
$u_2^{(n)} :=$ { solve $Lu_2^{(n)} = f_2$ in Ω_2
subject to
$u_2^{(n)} = g_2$ on $\partial\Omega_2 \bigcap \partial\Omega$
$\frac{\partial u_2^{(n)}}{\partial n_2} = -\partial u_1^{(n-1)}/\partial n_1$ on γ };
} until converge; } end.

The modified algorithms, like the original Schwarz alternating method, can be implemented on a massively parallel computer provided there exists data parallel direct solvers or data parallel iterative solvers. The efficiency of the modified algorithms on a coarse-grained machine depends on the load-balancing amongst the processing elements. Furthermore, new boundary conditions along the interface require data exchange as well as certain modification of the data. Therefore a slight overhead is introduced into the modified algorithms.

The two variants presented above is equivalent to solve the equation $F(u_\gamma) = 0$. If the differential operator L is linear, it is simply solving $F(u_\gamma) \equiv Cu_\gamma - f_\gamma = 0$, where C is the capacitance matrix as described previously which is a coupling matrix to couple the subdomains.

3.3. The generalized Schwarz alternating method

In order to allow more flexible treatment on the boundary conditions along the interface, the generalized Schwarz alternating method should be used. There are many different ways of imposing boundary conditions along the interface, just as there are many integral equations derived from potential theory.

The generalized Schwarz alternating method
for non-overlapped domain decomposition:
begin {
$n := 0$;
$\Phi[u_1^{(0)}], \Phi[u_2^{(0)}] :=$ initial approximation;
repeat {
$n := n + 1$;
$u_1^{(n)} :=$ { solve $Lu_1 = f_1$ in Ω_1
subject to
$u_1^{(n)} = g_1$ on $\partial\Omega_1 \bigcap \partial\Omega$
$\Phi[u_1^{(n)}] = \Phi[u_1^{(n-1)}] + \alpha\{\Phi[u_2^{(n-1)}] - \Phi[u_1^{(n-1)}]\}$ on γ };
$u_2^{(n)} :=$ { solve $Lu_2 = f_2$ in Ω_2
subject to

$u_2^{(n)} = g_2$ on $\partial\Omega_2 \bigcap \partial\Omega$
$\Psi[u_2^{(n)}] = \Psi[u_1^{(n-1)}]$ on γ };
} until converge; } end.

Numerically, such an exchange of information along the interface is equivalent to the set up of a capacitance system. An efficient numerical coupling of adjacent subdomains is equivalent to the determination of a suitable α in the above algorithms, and it is realized through suitable iterative methods, such as preconditioning conjugate gradient methods, and more recently fixed point iterations and quasi-Newton iteration (25).

3.4. Multi-subdomain

The Schwarz alternating method and its variants for the case of two-subdomain as discussed above can be extended to multi-subdomain simply by recursively substituting the two-subdomain algorithm into each of the subdomains. Such recursive substitution is also known as nested dissection (15).

For MIMD implementation with a fixed problem size, the number of subdomains increases as the subproblem size decreases. Direct methods to solve the subproblem becomes less expensive compare with iterative methods. Similar behaviour as that discussed in section 2.1 can be observed. For SIMD implementation with a fixed problem size, it is important to ensure that each subproblem can be mapped onto the SIMD environment.

4. CONCLUSIONS

The major concern of diakoptics was the numerical solution of some large scale scientific computations. Similarities of diakoptics and discrete representation of domain decomposition are discussed in terms of parallel implementations. The following suggestions are made. First, the above algorithms can be implemented on fine-grained machines with the provision of an efficient SIMD parallel solve routine and that the subproblems are solved in a serial fashion. Second, the above algorithms can be implemented on coarse-grained machines without much change to the original serial algorithm. However the number of subproblems is limited. Ideally, each processor will handle one subproblem, and hence to avoid idle time the subproblems should have similar computational work.

REFERENCES

[1] Schwarz, H. A. (1896) Über einige Abbildungsaufgaben. *Gesammelete Mathematische Abhandlungen*, **11**, 65–83.
[2] Fox, L. (1944) Solution by relaxation methods of plane potential problems with mixed boundary conditions. *Quarterly of Applied Mathematics*, **2**, 251–257.
[3] Kron, G. (1944) Equivalent circuits of elastic field. *Journal of Applied Mechanics*, **2**, A149–A161.
[4] Kron, G. (1945a) Equivalent circuits of compressible and incompressible fluid fields. *J. Aeronautical Sci.*, **12**, 221–231.
[5] Kron, G. (1945b) Numerical solution of ordinary and partial differential equations by means of equivalent circuits. *J. Applied Physics*, **16**, 172–186.
[6] Crank, J. and Nicolson, P. (1947) A practical method for numerical evaluation of solutions of partial differential equations of the heat-conduction type. *Proc. Cambridge Philosophical Society*, **43**, 50–67.
[7] Kron, G. (1963) *Diakoptics: The Piecewise Solution of Large-Scale Systems*. MacDonald & Co, London.
[8] Przemieniecki, J. S. (1963) Matrix structural analysis of substructures. *AIAA J.*, **1**, 138–147.
[9] Hockney, R. W. (1970) The potential calculation and some applications. *Methods Comput Phys*, **9**, 135–211.
[10] Buzbee, B. L., Dorr, F. W., George, J. A. and Golub, G. H. (1971) The direct solution of the discrete Poisson equation on irregular regions. *SIAM J. Numer. Anal.*, **8**, 722–736.
[11] Astrakhantsev, G. P. (1978) Fictitious domain method for second order elliptic equation with natural boundary conditions (in Russian). *Zh. Vychisl. Mat. i. Mat. Fiz.*, **18**, 119–125.
[12] Dryja, M. (1982) A capacitance matrix method for Dirichlet problem on polygonal region. *Numer. Math.*, **39**, 51–64.
[13] Golub, G. H. and Mayers, D. (1983) The use of preconditioning over irregular regions. *Sixth International Conference on Computing Methods in Applied Sciences and Engineering*, Versailles, France.
[14] Lai, C.-H. (1985) Application of DAP to computational aerodynamics. *PhD Thesis*, University of London.
[15] Wait, R. (1985) The solution of finite element equations on the DAP. *Technical Report*, University of Liverpool.
[16] Bjostad, P. E. and Widlund O. B. (1986) Iterative methods for the solution of elliptic problems on regions partitioned into substructures. *SIAM J. Numer. Anal.*, **23**, 1097–1120.
[17] Bramble, J. H., Pasciak, J. E. and Schatz, A. H. (1986) An iterative method for elliptic problems on regions partitioned into substructureds. *Math. Comp.*, **46**, 361–369.
[18] Keyes, D. E. and Gropp, W. D. (1987) A comparison of domain decomposition techniques for elliptic partial differential equations and their parallel implementation. *SIAM J. Sci. Stat. Comput.*, **8**, 166–202.
[19] Gentzsch, W., Neves, K.W. and Yoshihara, H. (1988) *Computational Fluid Dynamics: Algorithms & Supercomputers*. AGARD Report No. 311, AGARD.
[20] Bowden, K. (1990) Kron's method of tearing on a transputer array with an application to the solution of a distributed system. *Transputer Mail Shot*, **March**, 48–58.
[21] Lai, C.-H. (1990) A parallel algorithm for transonic flow calculations. *Appl. Math. Modelling*, **14**, 495–500.
[22] Trew, A. and Wilson, G. (1991) *Past, Present, Parallel: A Survey of Available Parallel Computing Systems*. Springer-Verlag, Berlin.
[23] Cross, M., Everett, M. G., Ierotheou, C. S., Johnson, S. P. and Jones, B. W. (1992) Parallel computational of CFD at the University of Greenwich. *Proc. Domain Decomposition for CFD on Message Passing Machines*, University of Leeds.
[24] Lu, T., Shih, T. M. and Liem, C.-B. (1992) *Domain Decomposition Methods—New Numerical Techniques for solving PDE*. Science Press, Beijing.
[25] Lai, C.-H. (1993) Comparing quasi-Newton methods for solving sparse interface problems. *CWI Technical Report*, NM-R9303. CWI, Amsterdam.
[26] Merugu, L.N. and Fusco, V.F. (1993) Concurrent network diakoptics for EM field problems. *IEEE Trans Microwave Theory & techniques*, **41**, 708–713.
[27] Bowden, K. (1994) The Kron approach to parallelization. In Marcer, P.J. and Fedorec, A.M. (eds), *Alternative*

Forms of Computation and New Routes to Parallelism. CASSM Publishing, University of Greenwich.
[28] Cross, M., Ierotheou, C. S., Johnson, S. P. and Leggett, P. F. (1994) CAPTools—semiautomatic parallelization of mesh based computational mechanics codes. *Presented at HPCN Europe*, Munich.
[29] Dekker, L., Smit, W. and Zuidervaart, J.C. (1994) *Massively Parallel Processing: Applications and Development*. Elsevier, Holland.